Pacotes Especiais Node JS

PROGRAMAÇÃO WEB 1

Objetivos de Aprendizagem

- Utilizar o gerenciador de pacotes NPM
- Identificar funções básicas do pacote Express

Agenda

- Instalação NodeJS
- *Node Package Manager*
- Módulos típicos
- *Express*

Node Package Manager

- Sistema de gerenciamento de módulos do NodeJS
- Similar ao `pip` do Python, Apt do Ubuntu, dentre outros
- Gerencia os pacotes (publica e instala) no sistema público
- Os instaladores do NodeJS geralmente realizam a instalação casada
- Para verificar a instalação e versões digite no console:
  - **`node -v`**
  - **`npm -v`**

Node Version Manager

- NodeJS disponibiliza ainda um utilitário de controle de versões
  - nvm (OSX e Linux)
  - nodist ou nvm-windows (Windows)
- Controladores de versões normalmente são instaladaos a parte
- Permite instalar e alternar entre diversas versões para testes
- O uso do nvm normalmente acontece via linha de comando

`npm init`

- Configuração inicial dos requisitos de um projeto Node JS novo ou existente
- Dentre outras ações cria um arquivo `package.json`
- O comando deve ser executado já na pasta do projeto, conforme exemplo

```
$ mkdir my-npm-pkg && cd my-npm-pkg
$ npm init
```

package.json

- Facilita o gerencimento dos pacotes necessários para a aplicação
- Lista todas as dependências de pacotes, inclusive versões
- Torna a estrutura da aplicação replicável, portanto fácil de manter

```
{
  "name": "npm-modules",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "lodash": "^4.17.21",
    "nodemon": "^2.0.15"
  }
}
```

package.json

- Ao executar `npm init` são solicitadas algumas informações básicas
  - **name:**
  - **version:**
  - **description:**
  - **scripts:**
  - **keywords:**
  - **author:**
  - **bugs:**
  - **homepage:**

```
{
  "name": "npm-modules",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "lodash": "^4.17.21",
    "nodemon": "^2.0.15"
  }
}
```

npm install

- Instala pacotes e suas dependências
- Quando executado sem parâmetros, conforme exemplo, instala as dependências listadas no arquivo `package.json`

```
$ npm install
```

npm install <module>

- Instala um pacote especificado e suas dependências
- Quando executado com a opção **-save**, conforme exemplo, instala o pacote escreve a dependência no arquivo **package.json**

```
$ npm install -save lodash
```

```
{
  "name": "npm-modules",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "lodash": "^4.17.21",
    "nodemon": "^2.0.15"
  }
}
```

npm install <module> -g

- Instala um pacote especificado não apenas para a aplicação corrente, mas sim globalmente
- Quando executado com a opção `-g`, conforme exemplo

```
$ npm install lodash -g
```

Resumo npm

Comando	Ação
`npm init`	Cria o arquivo **`package.json`** personalizado para um projeto
`npm init -y`	Cria o arquivo **`package.json`** padrão
`npm install`	Instala as dependências contidas no arquivo **`package.json`** do projeto
`npm install <package> -save`	Instala o pacote especificado para o projeto corrente e inclui a dependência no arquivo **`package.json`**
`npm install <package> -g`	Instala o pacote especificado como pacote padrão do Node JS

“Nodemon is a tool that helps develop node.js based applications by automatically restarting the node application when file changes in the directory are detected.”

Módulo nodemon

https://www.npmjs.com/package/nodemon

nodemon

- Não demanda não nenhuma modificação no código para sua utilização
- Substitui o comando node para inicializar as aplicações
- A aplicação é reiniciada a qualquer alteração no código
- Mesmo que a aplicação seja finalizada sem erros o monitoramento continua e será reinicializada a qualquer alteração

nodemon

Instalação

```
$ npm install -g nodemon
```

Utilização

```
$ nodemon [app_name]
```

“Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications.”

Módulo Express

https://expressjs.com/

Módulo Express

- *Framework* para desenvolvimento web com Nodejs bastante popular
- Provê um servidor web através do módulo http
- Utiliza-se de funções chamadas de *middlewares* para a criação das funcionalidades de um site ou aplicação
- Provê ainda a funcionalidade de *templates*

Gerador de aplicações

- O Express.js disponibiliza a ferramenta **`express-generator`** para a criação automatizada da estrutura de uma nova aplicação
- Para instalar o **`express`** e o **`express-generator`** basta utilizar o próprio **`npm`**

```
$ npm install express express-generator
```

Gerador de aplicações

- Instalados o **`express`** e o **`express-generator`** para criar uma aplicação basta executar os comandos mostrados em uma das caixas ao lado a partir do diretório da aplicação
- Em seguida instalar as dependências da aplicação com npm install
- Rodar a aplicação com npm start

```
$ npx express-generator [app_name]
$ npm install
$ npm start
```

```
$ npx express-generator --view=pug
[app_name]
$ npm install
$ npm start
```

Ferramentas

- https://replit.com/
- Visual Studio Code
- Web Storm (Jet Brains)

Exercício 1

1. Utilizando o Replit, fazer as seguintes tarefas:
    1. Utilizar o `npm` para criar `package.json` da aplicação pw1app na sua versão 1.0.0
    2. Instalar os pacotes:
        1. `nodemon`
        2. `http`
    3. No arquivo `index.js`, já existente, copiar o código do último exemplo da aula anterior
    4. Rode a aplicação. Verifique se está funcionando como esperado.
    5. Pare a aplicação.
    6. Utilize o `nodemon` para rodar a aplicação novamente
    7. Modifique o código fonte do servidor. O que acontece?

Exercício 2

2. Utilizando o Replit realize as seguintes ações:

1. Apagar todos os arquivos inseridos pelo Replit
2. Instale os seguintes pacotes usando o npm: `express` `express-generator` `nodemon`
3. Utilize o `express-generator` para criar a infra-estrutura básica para uma aplicação chamada `app`
4. Inicie a aplicação utilizando a linha de comando (node, npm ou nodemon)
5. O que acontece ao tentar iniciar a aplicação com o botão Run?
6. Verifique no arquivo `package.json` a seção *scripts*. Modifique o *script* `start` para que a aplicação seja iniciada com `nodemon`
7. Edite o arquivo /routes/index.js como mostrado a seguir
8. Edite o arquivo /views/indes.jade como mostrado a seguir
9. O que acontece com a aplicação?

routes > index.js > router.get('/') callback > ...

```
var express = require('express');
var router = express.Router();

/* GET home page. */
router.get('/', function(req, res, next) {
  res.render('index', { title: 'Express',
mysite: 'This is my first express site' });
});

module.exports = router;
```

views > index.jade

```
extends layout

block content
  h1= title
  p Welcome to #{title}
  p #{mysite}
```

Referências

- https://docs.npmjs.com/creating-a-package-json-file
- https://expressjs.com/pt-br/